**JavaScript 技术管理文档规范**

**强制（必须遵守）**

1. **禁止使用 var，统一使用 let 和 const。**
2. **所有语句必须以分号 ; 结尾，禁止省略。**
3. **使用 === 和 !==，禁止使用 == 和 !=。**
4. **禁止在非模块环境中污染全局变量。**
5. **函数必须有明确的返回值，避免隐式返回。**
6. **所有模块必须使用 import/export，禁止使用 require/module.exports。**
7. **禁止修改内建对象（如 Array.prototype、Object.prototype）。**
8. **禁止直接操作 DOM，应通过框架生命周期或指令操作。**
9. **禁止将业务逻辑写在事件绑定中，应封装成函数。**
10. **禁止魔法数字，需提取为常量命名。**
11. **函数参数不得超过 3 个，超出请封装为对象。**
12. **必须处理所有异步操作的异常（try-catch 或 .catch）。**
13. **禁止使用 eval、Function 构造器等动态执行字符串代码。**
14. **禁止使用 with 语句。**
15. **禁止未初始化的变量声明（如 let x; 必须尽快赋值或标明用途）。**
16. **禁止函数嵌套过深（最多 3 层）。**
17. **禁止写死 URL、路径、密钥等敏感信息。**
18. **所有项目使用统一 ESLint 配置进行代码质量校验。**
19. **禁止未授权跨域请求、需由服务端代理。**
20. **禁止使用同步 Ajax（如 XMLHttpRequest 同步模式）。**
21. **必须为循环语句添加退出条件，避免死循环。**
22. **必须在异步函数上显式标记 async，不可遗漏。**
23. **禁止使用过时 API，如 alert()、confirm()。**
24. **必须明确代码中每一处 this 的作用域归属。**
25. **文件必须使用 UTF-8 编码，使用 LF 换行。**

**推荐（尽量遵守）**

1. **优先使用 const 声明不可变变量。**
2. **使用模板字符串 ${} 替代字符串拼接。**
3. **变量/函数命名使用驼峰命名法（camelCase），类使用大驼峰（PascalCase）。**
4. **文件命名采用小写加中划线（kebab-case）。**
5. **每个文件只包含一个模块（组件、类或函数集合）。**
6. **每个函数控制在 50 行以内，超出应拆分。**
7. **模块入口文件命名为 index.js。**
8. **使用解构赋值提高可读性。**
9. **使用对象展开符（...）代替 Object.assign。**
10. **函数内部尽量少使用全局变量。**
11. **逻辑分支超过 3 层时使用早返回优化代码结构。**
12. **避免出现重复代码，应抽象出公共方法。**
13. **统一使用单引号 '，避免双引号 "。**
14. **模块导入顺序：内置模块 → 第三方 → 项目内部路径。**
15. **组件应添加注释说明功能与使用方式。**
16. **空对象建议写为 {} 而不是 new Object()。**
17. **避免不必要的依赖注入，应精简模块依赖。**
18. **逻辑清晰处可以使用三元表达式，复杂时应改为 if...else。**
19. **使用 Array.isArray() 判断数组类型，不使用 typeof。**
20. **使用 Number.isNaN() 替代全局 isNaN()。**
21. **使用 for...of 遍历数组，不使用 for...in。**
22. **确保组件/模块的单一职责。**
23. **使用命名函数提升调试可读性（避免匿名函数）。**
24. **尽量使用不可变数据结构（如使用 spread 拷贝数组）。**
25. **合理使用节流（throttle）和防抖（debounce）处理高频事件。**

**允许（可自由选择）**

1. **允许使用可选链 ?. 和空值合并操作符 ??，提高空值安全。**
2. **允许使用 console.log 进行调试，发布前需移除或封装为日志工具。**
3. **允许使用箭头函数简化回调，但注意作用域差异。**
4. **允许使用模块别名缩短导入路径（如 @/components）。**
5. **允许在逻辑简单场景中内联函数或表达式。**
6. **允许在文件中使用 // TODO 或 // FIXME 做标记，需定期清理。**
7. **允许使用 Lodash、Dayjs 等工具库，需控制依赖大小。**
8. **允许使用 CSS-in-JS 或样式模块，视项目结构选择。**
9. **允许使用生成器函数 function\*，适用于复杂流程控制。**
10. **允许使用 TypeScript 逐步替代 JavaScript，提升类型安全。**